TD 7 : CONCURRENCE

masquer=1

1. ISOLATION ET SÉRIALISABILITÉ

1.1 Soit transfert la procédure de transfert(X,Y,Z) entre comptes bancaires suivante :

procedure *transfert* (CompteDébité, CompteCrédité, montant)

*(1) variable := **Lire**(CompteDébité);*

*(2) **Ecrire** (CompteDébité, variable – montant);*

*(3) variable := **Lire** (CompteCrédité);*

*(4) **Ecrire(CompteCrédité, variable + montant);***

On lance simultanément les programmes transfert(A, B, 150) et transfert(B, C, 70). Chacun des processus (i.e., chacune des transactions de transfert) utilise une variable locale à son propre espace mémoire. Soient A_0, B_0, C_0 les valeurs initiales des trois comptes concernés.

1.1.1 Quel est l'état final des comptes A, B et C dans chacun des cas ci-dessous, où ABi représente l'ième instruction du transfert de A vers B et BCj la jème instruction du transfert de B vers C?

cas (a) = exécution complète de *transfert*(B,C,70), puis exécution de *transfert*(A,B,150)

cas (b) = exécution des instructions dans l'ordre

AB1 BC1 BC2 BC3 AB2 AB3 AB4 BC4

cas (c) = exécution des instructions dans l'ordre

AB1 AB2 AB3 BC1 BC2 BC3 AB4 BC4

1.1.2 Quelle contrainte d'intégrité n'est pas respectée par l'une des exécutions précédentes ?

1.1.3 Dans chacun des deux derniers cas, peut-on obtenir une exécution séquentielle des deux programmes de transfert, par permutation d'opérations successives non conflictuelles ?

1.1.4 Donnez dans chacun des cas (a), (b) et (c) de l'exercice précédent le *graphe de précédence* du transfert de A vers B, et du transfert de B vers C.

2. VERROUILLAGE EN DEUX PHASES

2.1 Soit la procédure *transfert* de l'exercice précédent dans laquelle on a inséré des instructions de verrouillage et de déverrouillage selon le protocole de *verrouillage en deux phases strict*.

2.1.1. Écrire la série d'instructions obtenue, en supposant que deux comptes différents appartiennent à des granules différents.

2.1.2. Les exécutions (b) et (c), décrites dans l'exercice précédent, sont-elles alors possibles ? Si non, donner une exécution possible ayant le même ordonnancement initial jusqu'à la première instruction non réalisable.

2.2 Soit la procédure *somme* qui affiche la somme des soldes de deux comptes bancaires :

procédure *somme* (Compte1, Compte2);

(1) var1 := ***Lire*** *(Compte1);*

(2) var2 := ***Lire*** *(Compte2);*

(3) Imprimer (var1 + var2);

où la procédure *Lire* recopie le solde du compte dans une variable en mémoire.

2.2.1. Écrire en SQL la série d'instructions (1) à (3).

2.2.2. *transfert* étant la procédure de l'exercice 1.2, on considère une exécution concurrente de *transfert(A,B,150)* et *somme(A,B)*, au cours de laquelle somme commence son exécution après les deux premières instructions de transfert. Dans le cas d'un verrouillage en deux phases, la procédure *somme* peut-elle s'exécuter en entier avant la reprise de *transfert* ?

2.2.3. Dans le cas d'un protocole qui lèverait le verrou posé par une transaction sur un granule dès que cette transaction aurait fini d'accéder au granule, la procédure somme pourrait-elle alors s'exécuter en entier avant la reprise de transfert ? Quel serait, dans ce cas, le résultat de l'exécution des deux transactions ?

2.3 Soit une base composée de quatre granules A, B, C, D et une exécution de six transactions T1 à T6 avec les accès suivants sur les granules :

A: E2(A) E3(A) L5(A)

B: L2(B) L4(B) L1(B)

C: E5(C) L1(C) L3(C) E4(C)

D: L6(D) L2(D) E3(D)

Ei(X) et Li(X) représentent une opération d'écriture (respectivement de lecture) du granule X par la transaction i.

2.3.1. Donner le graphe de précédence de cette exécution.

2.3.2. On suppose que le contrôle de concurrence est effectué par du verrouillage en deux phases strict, et que les demandesd'accès se font dans l'ordre suivant :

E2(A) L2(B) L6(D) E5(C) E3(A) L5(A) L1(C) L2(D) L3(C) E4(C) E3(D) L4(B) L1(B).

- Indiquer pour chaque granule les verrous en cours, et les demandes en file d'attente.

- Donner le graphe d'attentes et conclure.

2.4 Soient cinq transactions T1, T2, T3, T4 et T5 et quatre granules X, Y, Z, T. On considère l'exécution suivante :

L4(Z) L5(X) L3(Y) L4(Y) E5(X) E3(Y) L4(T) L3(Y) L3(Z) L4(Z) E4(T) L5(T) E1(T) E1(X) L2(Y) L4(Z) L3(X) E2(T)

2.4.1. Construire le graphe de précédence de l'exécution. Est-elle sérialisable ? Si oui, donner la ou les exécutions en série équivalente(s).

2.4.2. Cette exécution peut-elle être obtenue par un mécanisme de verrouillage en deux phases ?

2.5 Soient cinq transactions T1, T2, T3, T4 et T5 et quatre granules X, Y, Z, T. On considère l'exécution suivante :

L2(T) L3(Y) L3(X) L3(T) E2(T) E2(X) E3(Y) E5(X) L2(Y) L2(Z) E4(Z) E1(X) L1(Y) E1(Y) L4(T) L1(Z) E4(T)

2.5.1. En supposant que chaque transaction effectue une demande de verrou sur un granule juste avant d'essayer d'y accéder et que toutes les transactions respectent le protocole de verrouillage en deux phases, insérer les commandes de verrouillage et déverrouillage dans chaque transaction.

2.5.2. On suppose que les demandes d'accès arrivent dans l'ordre de l'exécution (ce qui ne veut pas dire que ces demandes seront satisfaites dans le même ordre). Indiquer comment se passe l'exécution des cinq transactions.

3. ESTAMPILLAGE

Soient les granules a, b, c et les transactions T1, T2, T3. On considère les exécutions suivantes:

a) L1(a), L1(b), E1(a), L2(b), E3(b), C3, E1(a), C1, L2(b), C2

b) L1(a), E2(a), C2, E1(a), C1, L3(a), C3

c) E1(a), L2(c), L2(a), C2, E1(a), L3(a), L1(c), C1, E3(b), C3

d) E1(a), L1(a), L3(b), E3(a), C3, L2(c), E2(a), E1(c), C1, L2(a), C2

On utilise un protocole de gestion de concurrence basé sur l'estampillage (l'estampille temporelle pour la transaction Ti est i). Pour chaque séquence montrer si l'estampillage sans la règle de Thomas et avec la règle de Thomas (lorsqu'elle est applicable) permet l'exécution de la séquence dans l'ordre donné. Montrer l'ordre d'exécution des opérations et l'ordre en série équivalent.

4. CLICHÉS MULTI-VERSIONS (FACULTATIF)

On suppose maintenant que le contrôle de concurrence utilise des clichés multi-versions. Cela permet de traiter les lectures sans demander de verrou partagé. Définir l'ordre d'exécution des opérations et l'ordre en série équivalent.

a)L1(a), L1(b), E1(a), L2(b), E3(b), C3, E1(a), C1, L2(b), C2

b) L1(a), E2(a), C2, E1(a), C1, L3(a), C3

c) E1a, L2c, L2a, C2, E1a, L3a, L1c, C1, E3b, C3

d) E1(a), L1(a), L3(b), E3(a), C3, L2(c), E2(a), E1(c), C1, L2(a), C2